Amendments to the Claims:

This listing of claims will replace all prior versions, and listing, of claims in the application:

Listing of the Claims:

1. (Currently Amended) A method of generating a Java macro instruction corresponding to one or more Java Bytecode instructions, said method comprising:

reading a stream of Java Bytecode instructions;

determining**, during Bytecode verification,** whether two or more Java Bytecode instructions in said Java Bytecode stream can be represented by one instruction;

generating**, at load time,** a Java macro instruction that represents said two or more Java Bytecode instructions when said determining determines that two or more Java Bytecode instructions in said Java Bytecode stream can be represented by one instruction, wherein said Java macro instruction is suitable for execution by a Java virtual machine;

generating**, at load time,** an internal representation of said Java macro instruction in a pair of streams that collectively represent an internal representation of said stream of Java Bytecode instructions in said Java virtual machine**, wherein said pair of streams consists of a code stream that is designated to store code associated with said Java macro instruction and a data stream that is designated to store data associated with said Java macro instruction**; and

wherein when executed**, at runtime,** said Java macro instruction can operate to perform one or more operations that are performed by said two or more Java Bytecode instructions.

2. (Original) A method as recited in claim 1,

wherein said determining operates to determine whether a predetermined sequence of two or more Java Bytecode instructions have been found.

3-6 (Cancelled)

7. (Currently Amended) A method of generating a Java macro instruction corresponding to one or more Java Bytecode instructions, said method comprising:

reading a stream of Java Bytecode instructions;

counting, during Bytecode verification, the number of times a sequence of Java Bytecode instructions appears in said stream of Java Bytecode instructions, said sequence of Java Bytecode instructions including two or more Java Bytecode instructions which are in a sequence in said stream;

determining, during Bytecode verification, whether said sequence of Java Bytecode instructions should be represented by one instruction;

generating, during Bytecode verification, a Java macro instruction that represents said sequence of Java Bytecode instructions when said determining determines that said sequence of Java Bytecode instructions can be represented by said one instruction;

representing said Java macro instruction in a pair of streams inside said virtual machine;

wherein said Java macro instruction is suitable for execution by a Java virtual machine; ~~and~~

wherein when executed**, at runtime,** said Java macro instruction can operate to perform one or more operations that are performed by said sequence of Java Bytecode instructions;

wherein said pair of streams includes a code stream and a data stream; and

wherein said code stream is designated to store a code portion of said Java macro instruction, and said data stream is designated to store a data portion of said Java macro instruction.

8. (Original) A method as recited in claim 7, wherein said determining of whether said sequence of Java Bytecode instructions should be represented by one instruction

operates to determine whether said sequence has been counted for at least a predetermined number of times.

9-13 (Canceled)

14. (Currently Amended) A method as recited in claim ~~13~~ **7**,

wherein said determining operates to determine whether a predetermined sequence of two or more Java Bytecode instructions have been found.

15. (Currently Amended) A method as recited in claim ~~13~~ **7**,

wherein said method further comprises counting the number of times a sequence of Java Bytecode instructions appear in said stream, and

wherein said determining operates to determine whether a sequence has been counted for at least a predetermined number of times.

16. (Currently Amended) In a Java computing environment, a Java macro instruction generator suitable for generation of Java macro instructions,

wherein each Java macro instruction corresponds to ~~one~~ **two** or more Java Bytecode instructions,

wherein said Java macro instruction generator operates to:

read a stream of Java Bytecode instructions during Java Bytecode verification;

determine**, during the bytecode verification,** whether two or more Java Bytecode instructions in said Java Bytecode stream can be represented by one instruction;

generate**, at load time,** a Java macro instruction that represents said two or more Java Bytecode instructions when said determining determines that two or more Java Bytecode instructions in said Java Bytecode stream can be represented by one instruction, wherein said Java macro instruction is suitable for execution by a Java virtual machine,

generate**, at load time,** an internal representation of said Java macro instruction in a pair of streams that collectively represent an internal representation of said stream of Java Bytecode instructions in said Java virtual machine**, wherein said pair of streams consist of a code stream that is designated to store code associated with said Java macro instruction, and a data stream that is designated to store data associated with said Java macro instruction**; and

wherein, when executed, said Java macro instruction can operate to perform one or more operations that are performed by said two or more Java Bytecode instructions.

17. (Original) A Java macro instruction generator as recited in claim 16, wherein said Java macro instruction generator operates during Java Bytecode verification.

18. (Original) A Java macro instruction generator as recited in claim 16, wherein said Java macro instruction generator operates to determine whether a predetermined sequence of two or more Java Bytecode instructions has been found.

19-20. (Canceled)

21. (New) A computer readable medium including computer program code for a Java macro instruction corresponding to one or more Java Bytecode instructions, said method comprising:

computer program code for reading a stream of Java Bytecode instructions;

computer program code for determining, during Bytecode verification, whether two or more Java Bytecode instructions in said Java Bytecode stream can be represented by one instruction;

computer program code for generating, at load time, a Java macro instruction that represents said two or more Java Bytecode instructions when said determining determines that two or more Java Bytecode instructions in said Java Bytecode stream can be represented by one instruction, wherein said Java macro instruction is suitable for execution by a Java virtual machine;

computer program code for generating, at load time, an internal representation of said Java macro instruction in a pair of streams that collectively represent an internal representation of said stream of Java Bytecode instructions in said Java virtual machine, wherein said pair of streams consists of a code stream that is designated to store code associated with said Java macro instruction, and a data stream that is designated to store data associated with said Java macro instruction; and

wherein when executed at runtime said Java macro instruction can operate to perform one or more operations that are performed by said two or more Java Bytecode instructions.

22. (New) A computer readable medium as recited in claim 21, wherein said determining of whether said sequence of Java Bytecode instructions should be represented by one instruction operates to determine whether said sequence has been counted for at least a predetermined number of times.

23 (New) A method as recited in claim 1, wherein said determining of whether said sequence of Java Bytecode instructions should be represented by one instruction operates to determine whether said sequence has been counted for at least a predetermined number of times.